

COEN-352

Tutorial #9

March 27th, 2023

Red-Black Trees

Red-Black tree is a **self-balancing binary search tree** that balances itself through its color property denoted by an extra bit.

Rules or Properties of Red-Black trees:

- 1. Root:** The root should be black.
- 2. Leaf:** Every NULL leaf (NIL) is black.
- 3. Red Nodes:** The child and parent of every red node is black.
- 4. Depth:** All the leaves have the same black depth (number of black nodes).
- 5. Path:** For each node, any simple path from this node to any of its descendant leaf has the same black-depth.

Attributes and Balancing Maintenance

Each node has the following attributes:

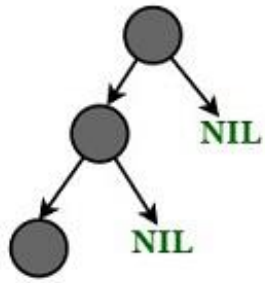
- **Color** (either **red** or black): *bool or int*
- **Key** or **Value**: *any type*
- **Left Child**: *Node*
- **Right Child**: *Node*
- **Parent** (except root node): *Node*

The limitations put on the node colors ensure that a simple path from the root to any leaf is not more than twice as long as any other such path. Thus the self-balancing property is maintained.

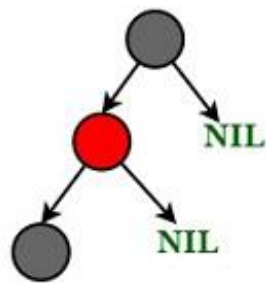
- The mechanism used is simple yet powerful.

Example: Red-Black Trees

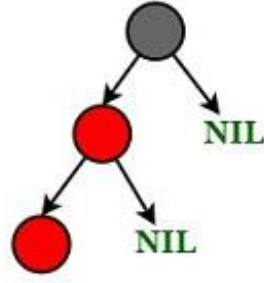
**Following are NOT possible
3-noded Red-Black Trees**



Violates
Property 4

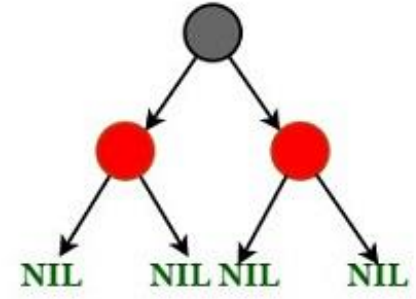
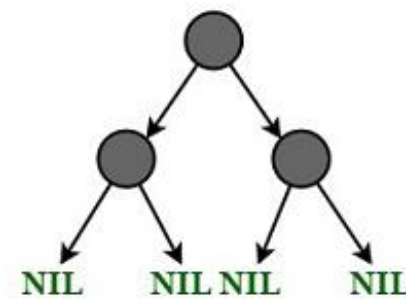


Violates
Property 4



Violates
Property 3

**Following are possible
Red-Black Trees with 3 nodes**



All Possible Structure of a 3-noded Red-Black Tree

Source: <https://www.geeksforgeeks.org/introduction-to-red-black-tree/>

Operations on Red-Black Trees: Rotations, Transplanting, Insertion, Deletion, Traversing, Searching

Two-Three Trees

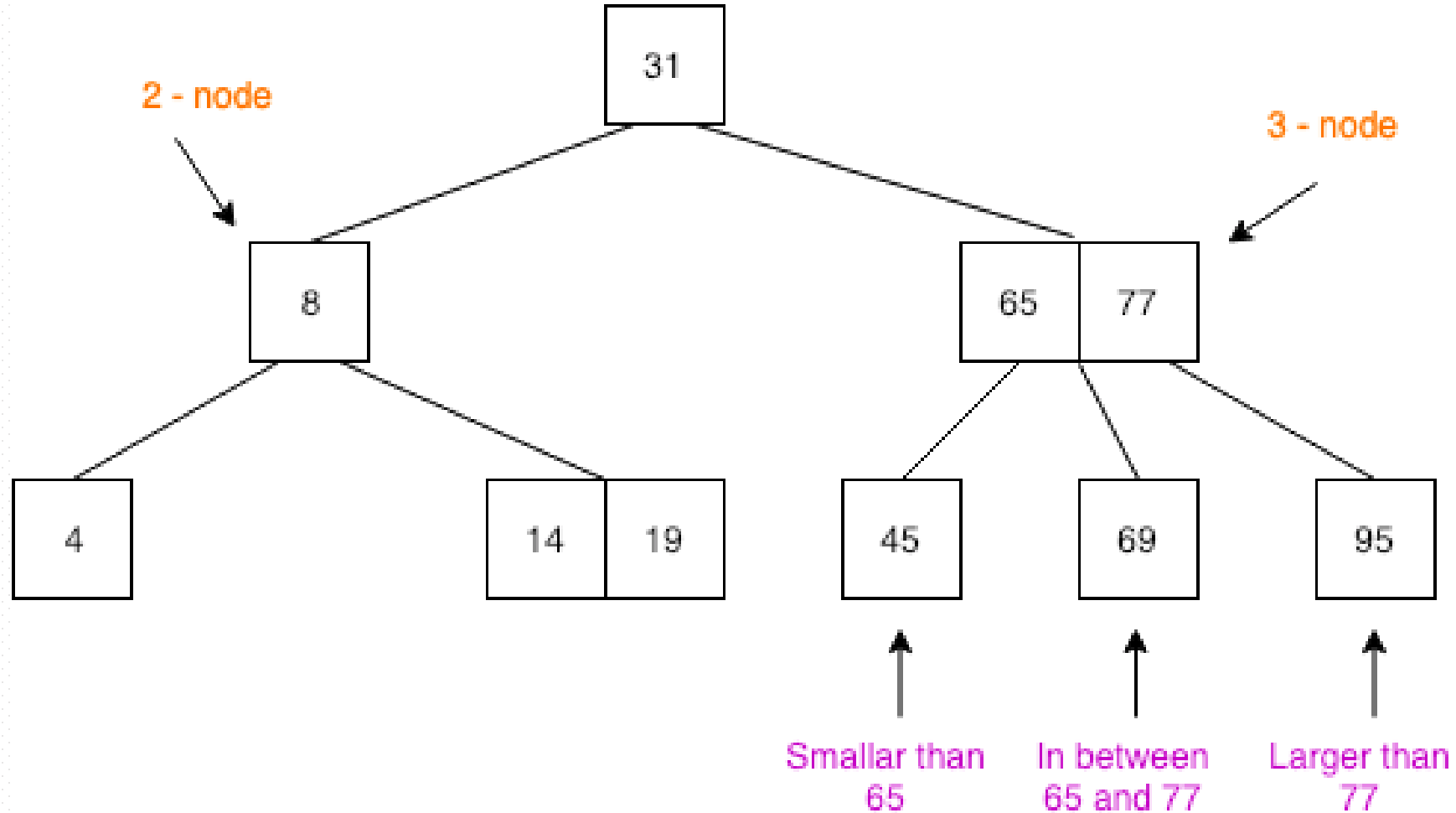
A **2-3 Tree** is a tree where a node can have 2 or 3 children nodes.

- A node with 2 children holds one key or value
- A node with 3 children holds 2 keys or values
- It follows the properties of a binary search tree
 - **For a 2-Children Node:** $\text{leftChild} < \text{Parent} < \text{RightChild}$
 - **For a 3-Children Node:** $\text{leftChild} < \min(\text{Parent}) < \text{middleChild} < \max(\text{Parent}) < \text{rightChild}$

A 2-3 tree maintains the following 2 properties:

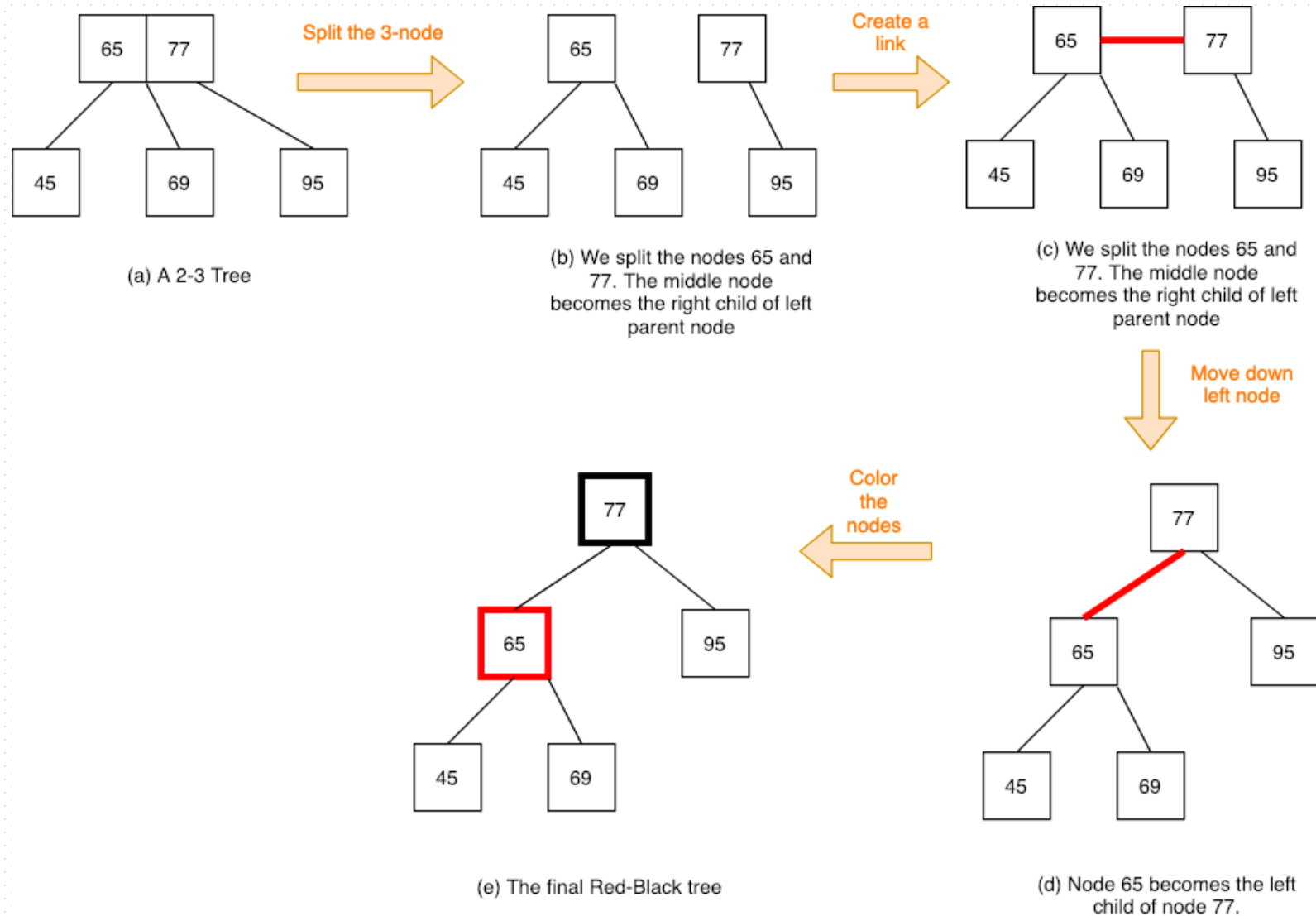
- **Perfect balance:** Every path from the root to the leaf node has the same length.
- **Symmetric order:** In-order traversal yields keys in ascending order (just like a binary search tree).

Example: Two-Three Tree



Source: <https://algorithmtutor.com/Data-Structures/Tree/2-3-Trees/>

Converting Red-Black Tree to 2-3 Tree



THANK YOU
